FFVVC: the VVC decoder in FFmpeg

Nuo Mi <nuomi2021@gmail.com>
Frank Plowman <post@frankplowman.com>
Shaun Loo <shaunloo10@gmail.com>

Agenda

- Introduction to FFVVC
  - VVC
  - FFVVC
- What’s new?
  - New coding tools
  - Thread model
- Performance
  - Versus other codecs
  - Versus VVC decoders
- GSoC
- Next steps

Disclaimer

Frank Plowman
frankplowman.com

- Introduction to FFVVC
  - VVC
  - FFVVC
- What’s new?
- Performance
- GSoC
- Next steps

H.265/VVC (Versatile Video Coding)

New standard from the JVET. Successor of H.264/AVC and H.265/HEVC.

Two objectives:

- 50% lower bitrates than HEVC
- Versatility:
  - Screen content coding
  - Adaptive resolution change
  - Independent subpictures

Open Source VVC

Encoders

- VTM
- VVenC
- uvg266

Decoders

- VTM
- VVdeC
- OpenVVC
- FFVVC

State of FFVVC

C merged at start of the year.

Inter prediction ASM merged.

Some other ASM in review.

Not yet Main-10 complete.

ASM Status (x86 only)

Module	C Decode Time	Reuse from HEVC	Complete	Priority
Intra	2.5%	maybe	0%	Low
Inter	18.5%	50%	50%	High
Transform	0.75%	10%	0%	High
LMCS	4%	0%	0%	Medium
Deblock	3.75%	50%	0%	High
SAO	2.5%	100%	100%	Medium
ALF	65%	0%	70%	High

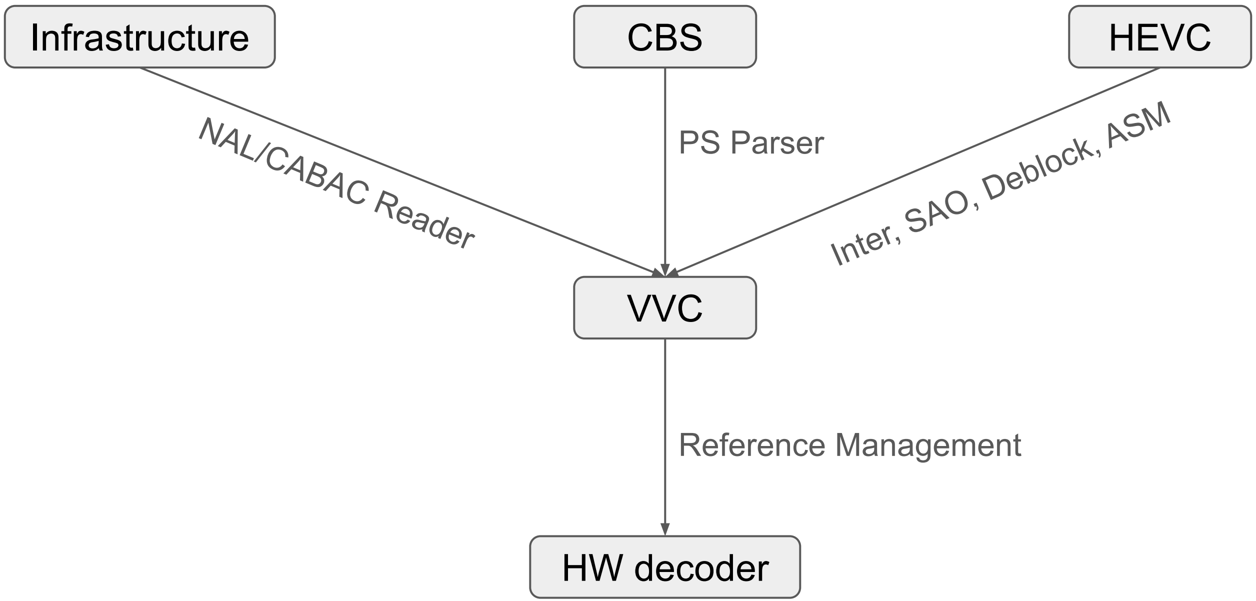
We need your help

Decoder size

Decoder	C (kLOC)	ASM (kLOC)
OpenVVC	47	167
VVdeC	49	12
FFVVC	18	—

Why FFVVC only needs 1/3 the code

We can reuse code or binary with others:



- Introduction to FFVVC
- What’s new?
  - New coding tools
  - Thread model
- Performance
- GSoC
- Next steps

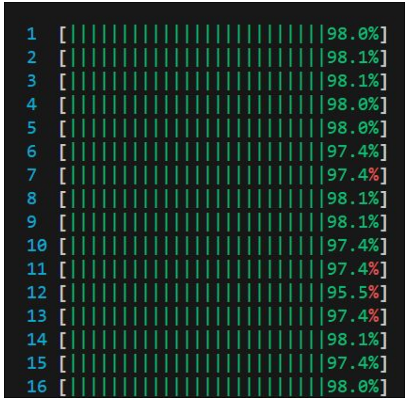
New coding tools

FFVVC has implemented a vast number of tools added to VVC

- Intra Prediction
  - Directional intra prediction modes
  - Cross-component linear model prediction
  - Position dependent intra prediction combination
  - Multiple reference line intra prediction
  - Intra Sub-Partitions
  - Matrix weighted Intra Prediction
- Inter Prediction
  - Extended motion vector prediction
  - Symmetric motion vector difference coding
  - Extended merge mode
  - Merge with motion vector difference
  - History-based Motion Vector Prediction
  - Affine motion compensated prediction
  - Subblock-based temporal motion vector prediction
  - Adaptive motion vector resolution
  - Motion field storage
  - Bi-prediction with CU-level weights
  - Bi-directional optical flow
  - Decoder side motion vector refinement
  - Geometric partitioning
  - Combined inter and intra prediction
- Transforms and Residual Coding
  - Integer Transforms and Quantization
  - Multiple transform selection
  - Subblock transforms for Inter CUs
  - Low frequency non-separable transform
  - Dependent quantization
  - Joint coding of chroma residuals
- Loop Filtering
  - Luma mapping with chroma scaling
  - Adaptive Loop Filter
- Versatile Coding Tools
  - Screen Content Tools (Todo)
  - 360◦ Tools (Todo)
  - Layered Coding (Todo)

Stage-based thread model

- FFHEVC has two thread models: frame and slice
  - They can not work together
  - No thread can cross frame or slice boundaries
- FFVVC has a more fine-grained thread model
  - Better able to utilise higher core counts
  - C code is able to decode 4k at over 30FPS using an i7-12700k

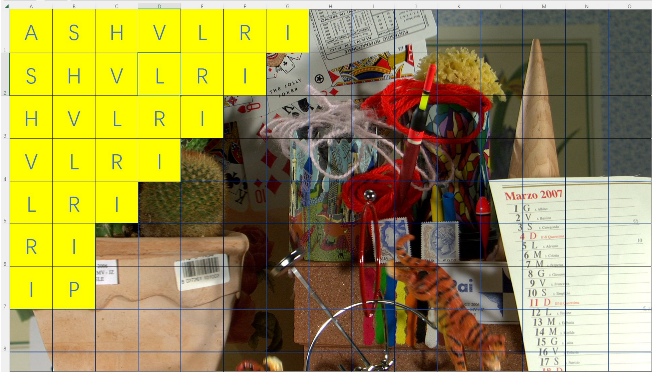


A 4k video decoding on 16 cores

Stage-based thread model

CTU divided into 8 stages:

- Parser (P)
- Inter (I)
- Recon (R)
- LMCS (L)
- Deblock V (V)
- Deblock H (H)
- SAO (S)
- ALF (A)



Informative, not correct

Each stage only depends on the current or previous stage of the neighboring CTUs.

Stage-based thread model

Facilitated by new `AVExecutor` utility

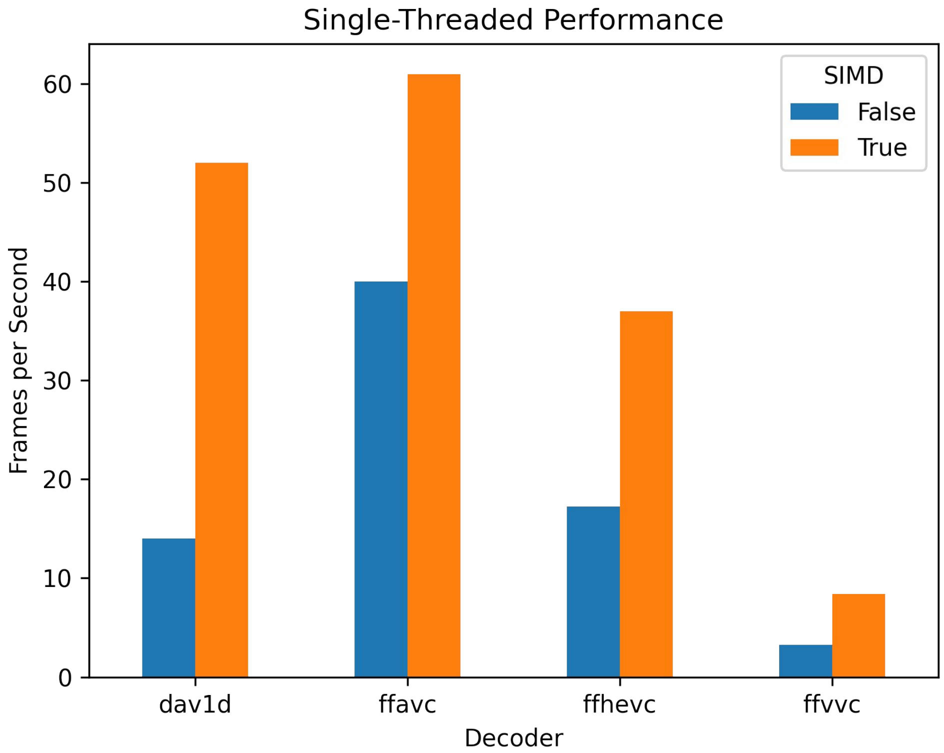
Tasks are put in a queue and scheduled to a thread

Simple algorithm, only 201 LOC and 1 real function

Made available in `libavutil`

- Introduction to FFVVC
- What’s new?
- Performance
  - Versus other codecs
  - Versus VVC decoders
- GSoC
- Next steps

FFVVC vs. FFHEVC vs. FFAVC. vs dav1d

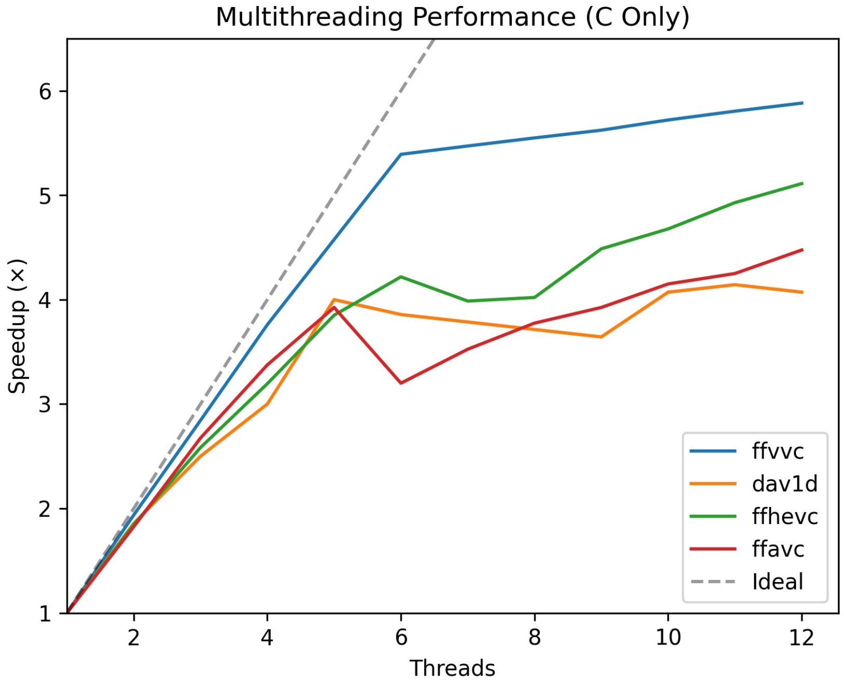


Netflix Sparks (natural content, 4096×2160, 10-bit, 4:2:0)

Encoded at 4MiB/s using VVenC, aom-av1, x264 and x265

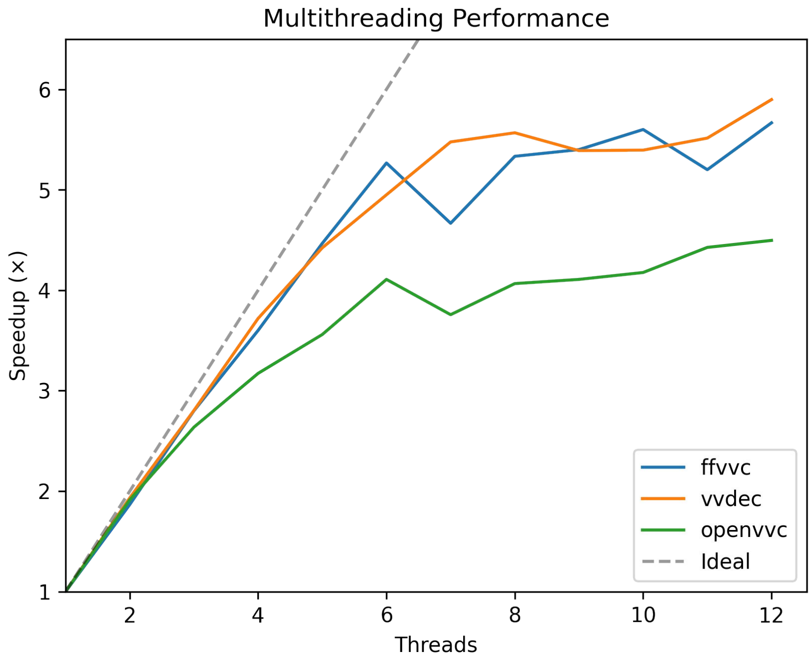
Decoded on an i7-8700K (AVX2, 3.70GHz)

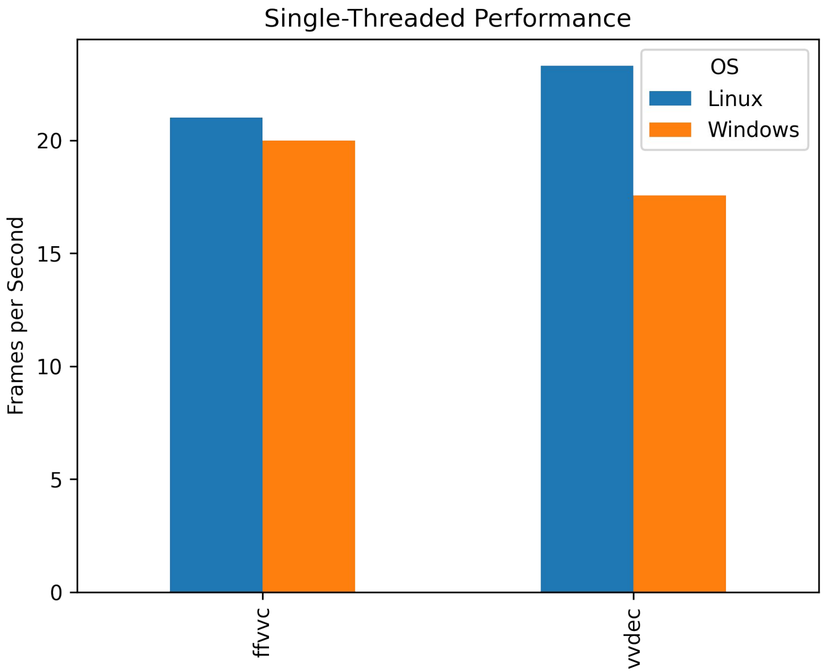
FFVVC vs. FFHEVC vs. FFAVC. vs dav1d



i7-8700K (6 cores, 12 threads)

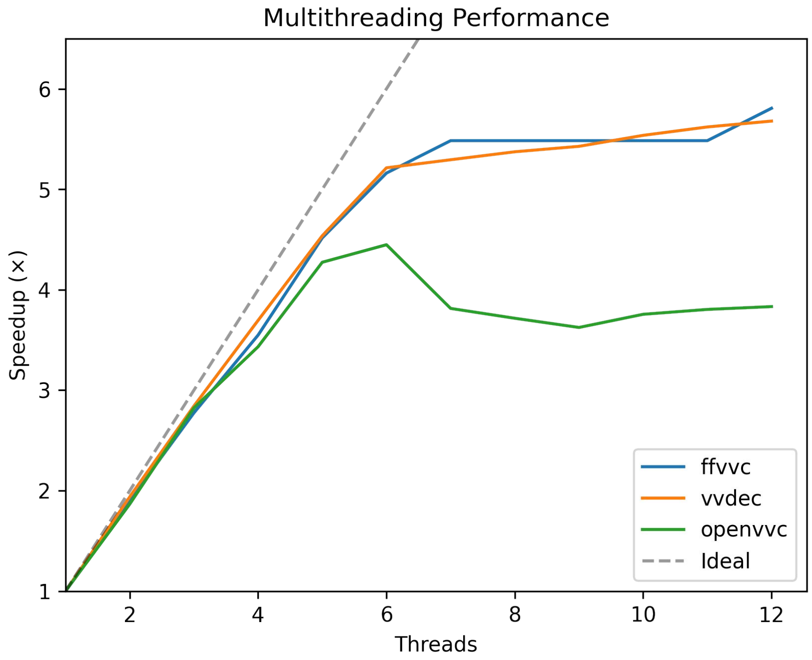
VVC Decoders 1080p Performance (C only)

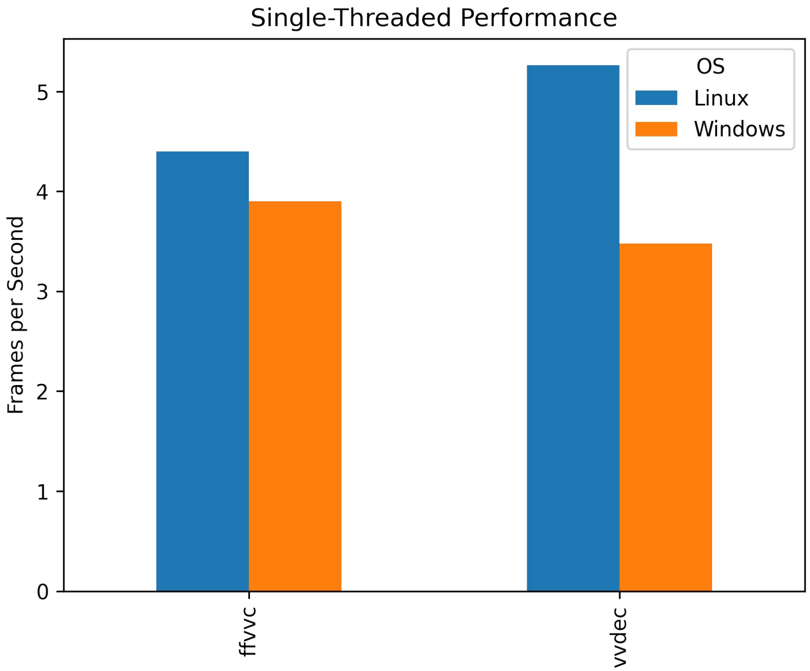




VVdeC is ~10% faster on Linux and ~10% slower on Windows.
Difference is in their single-threaded performance, their speedups are similar.

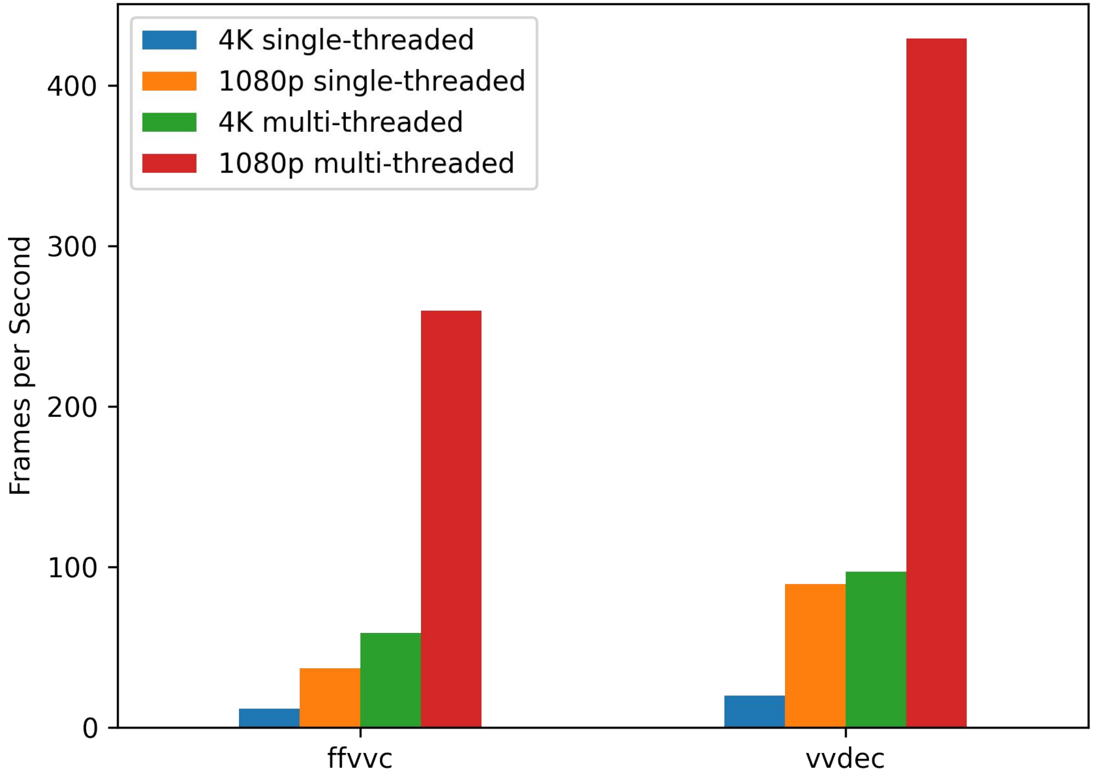
VVC Decoders 4K Performance (C only)





Story is similar yet slightly more pronounced for 4K.

1080P and 4k performance (ASM code)



VVdeC ASM is

90% faster single-threaded

75% faster multi-threaded.

- Introduction to FFVVC
- What's new?
- Performance
- GSoC
- Next steps

Google Summer of Code 2023

Frank Plowman

Implemented support for 12,14 bit-depths and range extension.
In-progress AVX-2 optimisations for inverse transforms

Shaun Loo

Implemented AVX-2 SAO, Deblock Chroma filters. Improvements for Deblock Chroma, Deblock Luma in-progress.

Next steps

- ASM
  - x86
    - Upstream existing code (ALF, SAO)
    - Implement more functions
      - Deblock
      - Transform
      - LMCS
  - ARM (GSoC 2024)
- New features
  - IBC, Palette and RPR
  - Thread optimization for 32+ cores
- GPU based decoder ?

Conclusion

FFmpeg now has its own VVC decoder.

It uses a codec parallelism technique new to FFmpeg.

C and multithreading performance is on par with VVdeC.

Optimised assembly is in the works.

Patches welcome!